Journal of Database Management, April-June, 2000, pp. 12 -- 19

Binary Equivalents of Ternary Relationships in Entity-Relationship Modeling: a Logical Decomposition Approach

Trevor H. Jones
School of Business Administration
Duquesne University
Pittsburgh, PA 15282

Il-Yeol Song
College of Information Science and Technology
Drexel University
Philadelphia, PA 19104

Abstract

Little work has been completed which addresses the logical composition and use of ternary relationships in entity-relationship modeling. Many modeling notations and most CASE tools do not allow for ternary relationships. Alternative methods and substitutes for ternary relationship structures do not necessarily reflect the original logic, semantics or constraints of a given situation. Furthermore, it has been shown that ternary relationships can be constrained by additional *implicit* binary constraints which do not occur in the logic of binary relationships.

This paper develops an analytical perspective of ternary relationships. We investigate the logical relationships implicit to the ternary structure and then identify potential simplification through decomposition into binary equivalents. These alternative binary equivalents allow retention of the implicit logical structure, and consequently also retain the semantics of the original structure. The analysis investigates equivalency of lossless decompositions, preservation of functional dependencies and finally the ability to preserve update constraints (insertions and deletions). We identify which ternary relationships have true, *fully equivalent*, binary equivalents and those which do not. We provide an exhaustive analysis of cardinality combinations found in ternary relationships which practitioners can use to guide the way in which they deal with ternary relationships in conceptual modeling.

Keywords: Entity relationship modeling, ternary relationships, functional dependency.

1 INTRODUCTION

One of the most widely used techniques in information analysis is the entity-relationship (ER) or extended entity-relationship model (EER, henceforth also referred to as ER), introduced by Chen (1976). While this technique is recognized as useful in its purpose, the form in which relationships are identified within such models still remains open to discussion. Certain arguments revolve around the inclusion of binary or N-ary representation of relationships in ER models. A central argument stems from the superior ability of N-ary modeling to reflect the true semantics of any given situation, whereas a binary model provides the simplest constructs for expressing information systems' logical design and is equivalently represented in a relational database management system (DBMS) (McKee & Rodgers, 1992).

The purpose of conceptual models is twofold: to provide a semantically correct, conceptual representation of the organizational data, as well as to provide a platform from which to develop a logical implementation schema. Consequently, the superior methodology for model construction is to adopt the semantically superior form and provide some heuristic set to allow transformation to a result, which can be implemented in the more, desired format. This course of action has been widely recognized and well researched, in the form of developing relational schema from ER diagrams; rule sets as well as automated tools have been developed which offer to guide the process of translation (e.g., Jajodia, 1983; Ling, 1985; Markowitz & Shoshani, 1992; Elmasri & Navathe, 1994).

Within these methodologies, there remains one area that has not been formally investigated. N-ary relationships in ER models continue to be constructs which are misunderstood by educators, difficult to apply for practitioners and problematic in their interpretation to relational schemas. These problems are due to causes ranging from a difficulty in identifying legitimate ternary relationships in practical situations to the lack of understanding of the construct in relation to the basis of normalization upon which the relational model is grounded. Song et al. (1995) provides a comparative analysis of conceptual modeling notations. While all of the notations had some allowance for ternary modeling, *none* of the CASE tools included in the study allowed for the use and translation of ternary relationships. This indicates the recognition of ternary relationships as having semantic significance, but a practical difficulty of implementing them beyond the equivalent logical level. Very little research has been completed on the theoretical underpinnings of N-ary relationships, and that which exists, is generally created as passing references in search of other research solutions.

This paper serves to provide insight to these problems. It seeks to formally analyze the dynamics of having three entities participating in a relationship simultaneously. This is done from two perspectives:

1. The theoretical approach to understanding this type of conceptual construct and the subsequent analysis of logical and relational models founded on these theories.

2. The practical aspects of using these constructs in entity-relationship modeling and how the various construct combinations can be mapped to the logical/physical model.

The second aspect is partly founded on the first because the potential decomposition of N-ary constructs, and their final representations, can be derived from theoretical analysis of the implicit relationships.

There will, therefore, be a particular effort to explain the simultaneous existence of N-ary and binary relationships that share the same participating entities and which are semantically related; this viewpoint has never been raised in previous research and leaves several questions unanswered. It is possible that a N-ary relationship may contain, or have imposed on it, a binary relationship between two of its participating entities which is semantically *related* to, and therefore potentially constrains, the ternary. Jones and Song (1996) have previously analyzed the question of which semantically related N-ary and binary relationship combinations can logically co-exist simultaneously. In their work, they have shown that only certain combinations of ternary/binary cardinalities may simultaneously co-exist and have provided a set of rules and notation that provide for conceptually correct modeling.

In providing an explanation of these implicit dynamics of N-ary structures, this work allows a further investigation of decomposition and restructuring of N-ary relationships to multiple binary structures, based on relational theory.

Many of the foregoing theoretical arguments lead to more utilitarian questions. The conceptual concept of the N-ary construct remains difficult for practitioners. Various notations exist for representing the construct in modeling, each with its own strengths and weaknesses. One of the most difficult problems is identifying exactly when a N-ary relationship should be used or when its binary equivalent is available. No prior work offering rules or heuristics can be found dealing with these questions. Typically, questions associated with ternary relationships are discussed within the context of 4NF and 5NF without regard for the specifically conceptual *modeling* problems. Since some of the solutions to these problems can be addressed with the previously mentioned theoretical background, each direction of analysis can contribute to the strength of the other, in terms of clarity and relevancy. This work seeks to address both the theoretical and practical issues surrounding N-ary relationships in conceptual modeling. Using a theoretical analysis of the construct, it seeks to provide practical answers to the understanding and use of those same constructs.

Terminology

Before proceeding to the substance of the paper, we first present certain terminology used throughout the paper. Since some terms in this field can be interpreted differently, this section provides a solid foundation for the ensuing discussions.

Ternary Relationship:

A ternary relationship is a relationship of degree three. That is, a relationship that contains three participating entities. Cardinalities for ternary relationships can take the form of 1:1:1, 1:1:M, 1:M:N or M:N:P. The cardinality constraint of an entity in a ternary relationship is defined by a pair of two entity instances associated with the other single entity instance. For example, in a ternary relationship R(X, Y, Z) of cardinality M:N:1, for each pair of (X, Y) there is only one instance of Z; for each pair of (X, Z) there are N instances of Y; for each pair of (Y, Z) there are M instances of X.

Semantically Constraining Binary (SCB) Relationship:

A Semantically Constraining Binary (SCB) relationship defines a binary constraint between two entities participating in a ternary relationship, where the semantics of the binary relationship are associated with those of the ternary and therefore affect potential ternary combinations of entity instances. They are differentiated from *Semantically Unrelated Binary* (SUB) Relationships, where a binary relationship exists between two entities that also participate in a ternary relationship but where the semantic of the binary relationship is unrelated to that of the ternary. A full explanation is provided in Jones and Song (1996). An example of this model type follows. We use the notation introduced by Jones and Song (1996) for the SCB relationships which consists of a broken line as opposed to a solid line used for the more common, independent, relationships.

Consider a ternary relationship between entities Teacher, Course and Section. The relationship has a cardinality of M:1:N respectively and models the sections associated with courses that teachers are currently involved with. Suppose we now wish to impose the constraint that a Teacher may only teach a single Course (which is *not* defined by the ternary relationship). The constraint is associated with the ternary relationship to the extent it restricts potential combinations. The modeling of this situation is shown in Figure 1. The binary relationship (A) between Teacher and Course would then be a SCB relationship. Notice that this model also shows an independent relationship (B) between Teacher and Course that might reflect, for example, which courses teachers *are capable* of teaching.

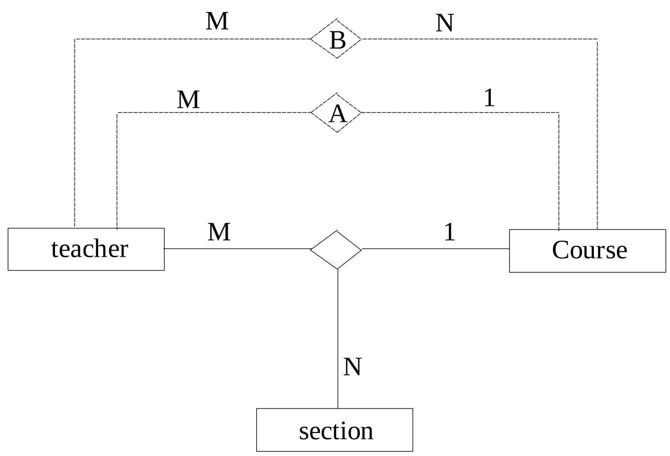

Figure 1. M:1:N Ternary Relationship with SCB Binary Relationship (A)

In considering functional dependencies within ternary relationships that have imposed binary relationships, we should remember that a binary functional dependency (FD) simply identifies an existence constraint within the ternary nature of the structure. The *minimal* determinant for a ternary relationship must be at least a composite of two entity identifiers.

2 DECOMPOSITIONS OF TERNARY RELATIONSHIPS

In this section, we discuss the decomposition of ternary relationships into binary relationships which are lossless, functional-dependency preserving and update preserving. Jones and Song (1993), in their analysis of ternary binary combinations have identified that if at least one binary constraint exists within a ternary relationship structure, the ternary relationship can be *losslessly* decomposed to a binary relationship structure. In this section we continue this line of reasoning because simple lossless decomposition is not sufficient to provide complete equivalency at the conceptual, subsequent logical and physical databases. We explore this question of equivalency and derive the allowable decompositions that provide *true* equivalencies.

The decomposition of any modeling structure assumes that the resulting structure(s) possesses all the implicit attributes of the original structure. That is, the alternatives are at least equal to the original structure, and *may* be more acceptable. When considering ternary relationships, we have identified three areas that we investigate to identify whether this equivalency is preserved. These are:

1. Whether the decomposition is lossless

2. Whether the decomposition preserves functional dependencies

3. Whether the decomposition can equivalently deal with the practical requirements of the constraint enforcement in the physical database resulting from the model (i.e. insertions, deletions), which we call update preservation constraints.

2.1 Lossless Decompositions

We begin by considering lossless decomposition strategies in two parts. In Jones and Song (1996) we find an analysis providing the fundamental basis for constraining ternary relationships through binary impositions (single and multiple constraints). Their analysis provides guidance on how the cardinalities of ternary relationships govern the allowance of *additional* constraints between the participating entities and the *potential* for further decomposition. From this analysis it is important to reiterate here, a fundamental rule, the IBC rule, which states:

"In any given relationship, regardless of ternary cardinality, the implicit cardinalities between any two entities must be considered M:N, provided that there are no explicit restrictions on the number of instances that can occur".

This means, fundamentally, that any *unconstrained* ternary relationship can not have an equivalent, binary decomposition structure (and also why unconstrained ternary relationships are not included in the analysis of decompositional equivalents in the remainder of the paper). We continue these arguments and review how certain cardinality combinations allow various decompositional outcomes. This review serves to maintain the context and initial considerations for considering the complete equivalency question between ternary relationships and potential binary decompositions.

Jones and Song (1993) have identified the various ternary/binary combinations and allowable decompositions. In providing the analysis they identify the Constrained Ternary Decomposition (CTD) rule which governs the potential lossless decomposition of ternary relationships. The CTD rule states:

"Any given ternary relationship cardinality can be losslessly decomposed to two binary relationships, provided that at least one 1:1 or 1:M constraint has been explicitly imposed between any two of the participating entities."

Table 1. Lossless Decompositions of Ternary Relationships

Case #	Ternary Cardinality (X:Y:Z)	Binary Imposition(s)	Potential Lossless Decomposition
1	1:1:1	(X:Y) = (M:1)	(XY)(XZ)
2	1:1:1	(X:Y) = (1:1)	(XY)(XZ) -or- (XY)(YZ)
3	1:1:1	(X:Y) = (M:1) (Z:Y) = (M:1)	(XY)(XZ) -or- (XZ)(ZY)
4	1:1:1	(X:Y) = (M:1) (X:Z) = (1:1)	(XY)(XZ) -or- (XZ)(ZY)
5	M:1:1	(X:Y) = (M:1)	(XY)(XZ)
6	M:1:1	(Y:Z) = (M:1)	(XY)(YZ)
7	M:1:1	(Y:Z) = (1:1)	(XY)(YZ) -or- (XZ)(ZY)
8	M:1:1	(X:Y) = (M:1) (Y:Z) = (1:1)	(XY)(YZ) -or- (XZ)(ZY) -or- (XY)(XZ)
9	M:1:1	(X:Y) = (M:1) (Y:Z) = (1:M)	(XZ)(ZY) -or- (XY)(XZ)
10	M:N:1	(X:Z) = (M:1)	(XY)(XZ)
11	M:N:1	(X:Z) = (M:1) (Y:Z) = (M:1)	(XY)(XZ) -or- (XY)(YZ)
12	M:N:P	Not Allowed	None

Table 1 provides an exhaustive summary of ternary/binary combinations and lossless decomposition strategies for the various cardinality combination outcomes. The outcome cardinalities shown represent all possible results that can be obtained through any allowable combination of binary/ternary cardinalities. We note that in accordance with the CTD rule, *all* constrained ternary structures have a potential lossless decomposition *except* M:N:P.

2.2 Functional Dependency Preserving Decompositions

When we consider the decomposition of a ternary relationship into a binary relationship equivalent, we are actually considering whether the equivalent model has the ability to represent, and enforce, all constraints that were present in the original structure. The desired constraints in entity relationship modeling are explicitly identified through the cardinalities associated with each relationship, or set of relationships. Consequently, we can test the equivalency of ternary formats against binary formats simply by comparing the implicit and explicit functional dependencies (which are derived from the cardinalities) found with each. Since the implicit cardinalities and functional dependencies may not necessarily be reflected in lossless decompositions, we should extend the investigation of decomposition to test whether functional dependency preservation is found in the lossless decompositions identified in Table 1.

We now complete this analysis and demonstrate that it is possible to have a lossless decomposition of a ternary relationship without the corresponding functional dependency preservation. The analysis of case #1, Table 1, for functional dependency equivalence follows.

Consider a ternary relationship R(X, Y, Z) with cardinality 1:1:1 and an imposed binary constraint of M:1 between X:Y. Using the notation identified previously, we can diagrammatically represent this structure as shown in Figure 2.

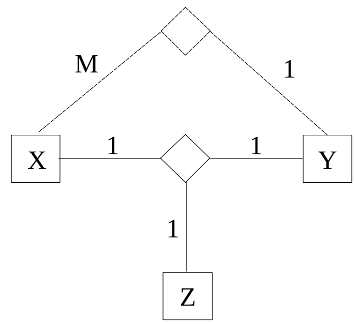

Figure 2. 1:1:1 Ternary Relationship with M:1 (XY) Constraint

The explicit constraint between X/Y implicitly suggests the following set of functional dependencies:

$XY \rightarrow Z$, $XZ \rightarrow Y$, $YZ \rightarrow X$, $X \rightarrow Y$, $X \rightarrow Z$, ($X \rightarrow Z$ is implicitly derived from $X \rightarrow Y$ and $XY \rightarrow Z$. The relationship Y:Z remains M:N (IBC rule).)

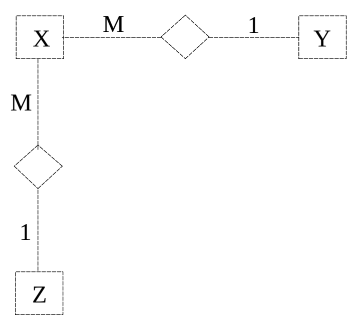

Figure 3. Suggested decomposition for Figure 2

According to Table 1, and the CTD rule, we may use a binary decomposition of (XY)(XZ), as shown in Figure 3.

Also consider a set of instances complying with the same cardinality requirements (Example 1):

X	Y	Z
X1	Y1	Z1
X2	Y1	Z2
X3	Y2	Z1

Example 1. Ternary Relationship Instance Set

This relation and population may be losslessly decomposed to (Example 2):

X	Y
X1	Y1
X2	Y1
X3	Y2

X	Z
X1	Z1
X2	Z2
X3	Z1

Example 2. Decomposed Ternary Storage Structure

As previously identified (Table 1), this decomposition is lossless. But are the two storage structures equivalent from the perspective of FD preservation? Consider the same decompositional scenario from a functional dependency perspective. We have previously defined the set of functional dependencies that were present in the original ternary structure. In the decomposed structure, we identify the following functional dependency set:

$X \rightarrow Y$, $X \rightarrow Z$, $XY \rightarrow Z$ (augmentation), $XZ \rightarrow Y$ (augmentation)

We note that in this decomposition strategy, we lose the functional dependency of $YZ \rightarrow X$, and there is no way to recover it through reconstruction based on Armstrong's Axioms (Armstrong, 1974). This supports the observation that a binary decomposition is not always able to enforce functional dependency constraints, even though the decomposition may be lossless.

Table 2. Lossless *and* FD Preserving Decompositions

Case #	Ternary Cardinality (X:Y:Z)	Binary Impositions	Potential Lossless Decomposition	Potential FD Preserving Decompositions
1	1:1:1	(X:Y) = (M:1)	(XY)(XZ)	None
2	1:1:1	(X:Y) = (1:1)	(XY)(XZ) -or- (XY)(YZ)	(XY)(XZ) -or- (XY)(YZ)
3	1:1:1	(X:Y) = (M:1) (Z:Y) = (M:1)	(XY)(XZ) -or- (XZ)(ZY)	(XY)(XZ) -or- (XZ)(ZY)
4	1:1:1	(X:Y) = (M:1) (X:Z) = (1:1)	(XY)(XZ) -or- (XZ)(ZY)	(XY)(XZ) -or- (XZ)(ZY)
5	M:1:1	(X:Y) = (M:1)	(XY)(XZ)	(XY)(XZ)
6	M:1:1	(Y:Z) = (M:1)	(XY)(YZ)	None
7	M:1:1	(Y:Z) = (1:1)	(XY)(YZ) -or- (XZ)(ZY)	(XY)(YZ) -or- (XZ)(ZY)
8	M:1:1	(X:Y) = (M:1) (Y:Z) = (1:1)	(XY)(YZ) -or- (XZ)(ZY) -or- (XY)(XZ)	(XY)(YZ) -or- (XZ)(ZY)
9	M:1:1	(X:Y) = (M:1) (Y:Z) = (1:M)	(XZ)(ZY) -or- (XY)(XZ)	(XZ)(ZY)
10	M:N:1	(X:Z) = (M:1)	(XY)(XZ)	(XY)(XZ)
11	M:N:1	(X:Z) = (M:1) (Y:Z) = (M:1)	(XY)(XZ) -or- (XY)(YZ)	None
12	M:N:P	Not Allowed	None	None

In scrutinizing all potential cardinality outcomes, and applying the same functional dependency and decomposition analysis, we find that *three* additional combinations (cases 1, 6, 11) cannot be decomposed without losing some level of functional constraint enforcement. Table 2 shows all possible combinations together with their decompositional status regarding functional dependency preservation. This table is then an extension of Table 1, where only the join and losslessness were considered and every combination qualified. We also notice in comparing the two sets of outcomes, that some of the alternate decompositions allowed in Table 1, do not preserve all functional dependencies and are therefore disqualified in the more restrictive Table 2.

2.3 Insertion / deletion constraint preservation

So far, in investigating dependencies within these alternate structures, we have considered only the ability to decompose *static* structures from a lossless and FD preserving perspective. That is, we have looked only at the ability to preserve a *specific set* of instances during decomposition and re-composition. We have not considered the ability of a binary model (as opposed to a ternary structure) to equivalently handle insertions and deletions of ternary relationships (dynamic decompositions), which may be reflected in the preservation of functional dependencies. We have simply identified that certain static structures have equivalent lossless forms. The additional consideration of updates, which are of significant importance to creating realistic database models, are investigated in this section.

2.3.1 Insertion Constraints

Let us consider the way the alternative storage structures (ternary v.'s binary) allow insertion of similar tuples. We should keep in mind that we are comparing the ability of the structures to ensure enforcement of all constraints present (typically identified by the implicit functional dependencies).

Consider the act of requesting an insertion of tuple R1(X4, Y1, Z1) into the database. In the original format of structuring the ternary relationship (Figure 2, Example 1 and case #1, Table 2), this insertion would be disallowed because it violates the constraint that a pair of YZ values must be associated with a single value for X via FD Y1, Z1 $\rightarrow$ X1. In the decomposed structure (Figure 3 and Example 2) the tuple would be decomposed and inserted in the form (X4, Y1) and (X4, Z1). These insertions into the binary equivalents would be accepted without violating any constraints of the binary structures. Checking the constraints YZ $\rightarrow$ X in the binary equivalents requires joins between two decomposed tables. We therefore have an obvious difference between the way the two structural representations would allow insertions.

The discussion here deserves two important observations. The first issue concerns the recognition and identification of a constraint at schema level. That is, while in the ternary format the functional dependency YZ $\rightarrow$ X can be observed and identified at the schema level, thus improving the modeling power; in the binary equivalents this constraint is not easily observed and identified. There is less opportunity for designers to identify the constraint in the binary equivalents. The second aspect concerns the enforcement of the constraint. Checking the constraint YZ$\rightarrow$ X does not require a join in the ternary structures, but requires a join in the binary equivalents. The binary equivalent format could cause inconsistency during insertion if not checked at all or degrade performance when checking the constraint using a join.

The conclusion is that not every ternary relationship can be decomposed to binary equivalents without losing the constraint and modeling power of original ternary relationships.

2.3.2 Deletion Constraints

Let us now move on to consider the same arguments with respect to binary alternatives correctly providing for deletion of tuples. Consider the ternary relationship R(X, Y, Z) with cardinality M:1:1 (case #8), and binary impositions M:1 (X, Y) and 1:1 (Y, Z). According to Table 2 this relation can be lossless and FD preserving, decomposed to its binary equivalent of S(X, Y) and T(Y,Z). An example of the original table and decomposition, with appropriate instances is shown in Example 3 and Example 4, below. Example 3 shows data as stored in the structure derived from a ternary relationship. Example 4 shows the data stored in the decomposed, functional dependency preserving alternative structure (see also Table 2). If we now attempt to delete tuple R_1(X1, Y1, Z1), the ternary storage structure dictates that we have tuples R_2(X2, Y1, Z1) and R_3(X4, Y3, Z2) remaining. However, if we delete the same semantic combination of data, R_1(X1, Y1, Z1), from the equivalent binary storage format, we must delete tuples S_1(X1, Y1) and T_1(Y1, Z1), based on the projections of R_1. We have now deleted the only reference to T_1 (Y1, Z1). This tuple is in fact required to reconstruct the remaining relationship R_2(X2, Y1, Z1). In other words, after the binary deletions, we are unable to reconstruct the expected tuple R_2(X2, Y1, Z1).

Another simple example helps to underscore this point. Let us assume we wish to delete tuple R3(X4, Y3, Z2). In Example 3, this is done by removing the complete tuple, with no additional consideration to any associated affects on the data. In the decomposed form, the same tuple is stored in its binary components. In this format however, we cannot simply delete the two binary projections of R_3, without first checking to see if either is required by additional binary components elsewhere in the schema. That is, we cannot delete (X4, Y3) from Relation S, without first checking to see if *multiple* values of Y3 exist in Relation T, not just the tuple (Y3, Z2). In our example here, since there is only a singular reference to Y3 as a foreign key, tuple (X4, Y3) could be deleted. However, if there were additional references to Y3 in Relation T, tuple (X4, Y3) *could not* be deleted as it would indicate that an additional ternary relationship tuple, consisting of (X4, Y3, Z#), may be required, and could not be reconstructed without the preservation of (X4, Y3).

Relation_R

X	Y	Z
X1	Y1	Z1
X2	Y1	Z1
X4	Y3	Z2

Example 3. M:1:1 Ternary Relationship with Binary Impositions

Relation_S

X	Y
X1	Y1
X2	Y1
X4	Y3

Relation_T

Y	Z
Y1	Z1
Y3	Z2

Example 4. Lossless & FD Preserving Decomposition of Example 3

We therefore observe that, although the decomposition is equivalent from the point of views of lossless and FD-preserving constraints, from a practical standpoint, the binary structure does not always have the implicit semantic constraint checking properties of the ternary structure in the case of deletions. We have no ability to preserve the necessary binary relationships without some additional mechanism by application programs to check reference requirements between tables. Additionally, even if we were to use additional constraint checking, since the original ternary relationship data now exists only as a set of binary combinations, we have no reference point to verify that any other ternary combinations *should* exist. For example, in the above case, how do we know in the future that tuple R_1 may even need to be reconstructed? The binary structure does not automatically provide the storage requirements to allow deletions without potential loss of data or semantic integrity. In fully analyzing which ternary/binary cardinality combinations fully preserve constraint checking on deletions we find there are a total of 8 cases (see Table 3 below).

Table 3. Implicit Semantic Constraint Enforcement

Case #	Ternary Cardinality (X:Y:Z)	Binary Impositions	Potential Lossless Decomposition	Potential FD Preserving Decomposition	Enforces Semantic Constraints on Insertions	Enforces Semantic Constraints on Deletions
1	1:1:1	(X:Y) = (M:1)	(XY)(XZ)	None	No	No
2	1:1:1	(X:Y) = (1:1)	(XY)(XZ) -or- (XY)(YZ)	(XY)(XZ) -or- (XY)(YZ)	Yes	Yes
3	1:1:1	(X:Y) = (M:1) (Z:Y) = (M:1)	(XY)(XZ) -or- (XZ)(ZY)	(XY)(XZ) -or- (XZ)(ZY)	Yes	Yes
4	1:1:1	(X:Y) = (M:1) (X:Z) = (1:1)	(XY)(XZ) -or- (XZ)(ZY)	(XY)(XZ) -or- (XZ)(ZY)	Yes	Yes
5	M:1:1	(X:Y) = (M:1)	(XY)(XZ)	(XY)(XZ)	Yes	Yes
6	M:1:1	(Y:Z) = (M:1)	(XY)(YZ)	None	No	No
7	M:1:1	(Y:Z) = (1:1)	(XY)(YZ) -or- (XZ)(ZY)	(XY)(YZ) -or- (XZ)(ZY)	Yes	No
8	M:1:1	(X:Y) = (M:1) (Y:Z) = (1:1)	(XY)(YZ) -or- (XZ)(ZY) -or- (XY)(XZ)	(XY)(YZ) -or- (XZ)(ZY)	Yes	No
9	M:1:1	(X:Y) = (M:1) (Y:Z) = (1:M)	(XZ)(ZY) -or- (XY)(XZ)	(XZ)(ZY)	Yes	No
10	M:N:1	(X:Z) = (M:1)	(XY)(XZ)	(XY)(XZ)	Yes	No
11	M:N:1	(X:Z) = (M:1) (Y:Z) = (M:1)	(XY)(XZ) -or- (XY)(YZ)	None	No	No
12	M:N:P	Not Allowed	None	None	No	No

2.4 Exhaustive analysis

Table 3 below (which extends Table 2) provides an exhaustive analysis of all ternary binary combinations and the full results of which are lossless, FD preserving, and update preserving with insertion and deletion constraints. We observe from this analysis that given all combinations of binary/ternary cardinality combinations, only four of the combinations and subsequent potential binary decompositions provide losslessness, functional dependency preservation, and update constraint (insert and delete) preservation. Consequently, only these four structures can be considered *fully* equivalent to the structure derived from the ternary relationship.

3 CONCLUSION

Ternary relationships are an established and accepted construct within the realm of (N-ary) conceptual modeling. However, very little research has been completed which investigates the logical complexity of the construct, particularly when combined with additional, internal constraints, which are termed SCB relationships. There is a valid question of whether ternary constructs, which have been semantically constrained, have fully equivalent binary decompositions. We have shown in this paper, that logical (fully equivalent) decompositions do exist for certain combinations of ternary / binary cardinalities, but the majority do not have fully (logical and practical) equivalents.

This work, in addition to providing a basis for manipulating ternary relationships, provides important facts concerning the implicit nature of the constructs, and therefore offers key insights into methodologies and mechanisms for dealing with them. If we accept that ternary relationships are “absolute”, that is they possess some singular representation in data modeling, then true equivalents of the structure must follow those derivations provided by the contents in this paper. We have shown that certain constraint combinations have no correct decomposition from a theoretical or practical perspective. We therefore deduce that arguments concerning ‘alternative’ representations of such constructs are invalid or contain additional and superfluous overhead required to represent the inherent properties of the original N-ary form. In binary modeling, the creation of 'entities', with composite keys ("gerunds", which substitute for ternary relationships), allow only partial representation of all possible semantics, since they do not allow graphical modeling of the intra-entity binary constraints (SCB relationships) which may be present within a ternary relationship. Furthermore, we have demonstrated that the simple preservation of functional dependencies in a decomposition does not always support the valid join of two projections if some level of update is involved between the decomposition and join (i.e. a join of two projections which preserve functional dependencies is only valid from an instance level, but cannot be extended to the database's implicit constraint level).

Although the arguments concerning insertions and deletions may be categorized as view update problems, we maintain that view update problems are derived from the creation of logical and physical aspects (subsets) of a database. What we suggest here is that the logical interpretation of the conceptual model is affected by the physical structure chosen in the final design. This need for translation reflects problems of maintaining functional dependencies between attributes in a model but potentially destroying the semantics of a particular (ternary) structure. We have demonstrated that the same sets of constraints that are associated with the semantics are not necessarily reflected in the (theoretical) preservation of the functional dependencies.

In conclusion, we suggest that if indeed the ternary relationship is considered a bone fide structure, modeling notations, and any derivations of the conceptual models containing ternary relationship structures, must recognize and fully implement the correct logic and semantics implicit to these types of structures. This includes modeling notations and the heuristics built into automated (CASE) tools. Towards this objective, this paper has identified the practical and theoretical underpinnings for this position and provides a decompositional framework (Table 3) to assist practitioners who may have to deal with these issues.

4 REFERENCES

Armstrong, W.W. (1974) "Dependency Structures of Data Base Relationships". In Proc. IFIP Congress, 1974.

Chen, P.P. (1976) The entity-relationship model: towards a unified view of data. ACM Transactions on Database Design, 1(1), pp. 9 -- 36.

Elmasri, R., & Navathe, S. (1994). Fundamentals of Database Systems. 2nd Edition, Benjamin/Cummings, Redwood City, CA. 1993.

Jajodia, S., Ng, P., & Springsteel, F. (1983) The problem of equivalence for entity-relationship diagrams. IEEE Transactions on Software Engineering, v. SE-9(5), September, 1983, 617 -- 630.

Jones, T.H. & Song, I.-Y. (1993). "Binary Imposition Rules and Ternary Decomposition". In Proceedings of InfoScience '93, Conference on Information Science and Technology, Korea Information Science Society, Seoul, Korea, October 21 -- 23, 1993, pp. 267 -- 274.

Jones T. & Song I.-Y. (1996). Analysis of Binary / Ternary Cardinality Combinations in Entity-Relationship Modeling. Data and Knowledge Engineering, 19 (1996), pp. 39 – 64.

Ling, T.W. (1985). " A Normal Form for Entity-Relationship Diagrams". In Proceedings of 4th International Conference on the Entity-Relationship Approach (Chicago). IEEE Computer Society Press, Silver Springs, MD., 1985.

Markowitz, V.M. & Shoshani, A. (1992). "Representing Extended Entity-Relationship Structures in Relational Databases: A Modular Approach". ACM Transactions on Database Systems, 17(3), September, 1992, pp. 423 -- 464.

McKee, R.L. & Rodgers, J.(1992). "N-ary Versus Binary Data Modeling: a Matter of Perspective". Data Resource Management, 3(4), Fall 1992, pp. 22 -- 32.

Song, I.-Y., Evans, M. & Park, E.K. (1995) A Comparative Analysis of Entity-Relationship Diagrams. Journal of Computer & Software Engineering, 3(4), 427 – 459, 1995.

Teorey, T.J. Database Modeling and Design: The E-R Approach, 3rd Ed. Morgan-Kauffman, 1999.